Package 'TockyDevelopment'

November 27, 2024

Version 0.1.0

Date 2024-11-19

Title Multidimensional Analysis and Pathfinding Analysis of High-Dimensional Flow Cytometric Fluorescent Timer Data

Author Masahiro Ono [aut, cre]

Maintainer Masahiro Ono <m.ono@imperial.ac.uk>

Depends R (>= 4.2.0), utils, stats, graphics, grDevices, methods

Imports igraph, ggplot2, RColorBrewer, bigmemory, biganalytics, vegan, TockyLocus, TockyPrep

Suggests knitr, rmarkdown

VignetteBuilder knitr

Description This package offers functions for analyzing the temporal dynamics of developing cells during development, utilizing Fluorescent Timer protein reporters, cross-analyzing Timer data and multidimensional marker expression profiles. The package enables identification of time-dependent trajectories using Fluorescent Timer data.

URL https://github.com/MonoTockyLab/TockyDevelopment

BugReports https://github.com/MonoTockyLab/TockyDevelopment/issues

Encoding UTF-8

RoxygenNote 7.3.2

Contents

`BiplotCCA` *Generate CCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)*

Description

Generate CCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)

Usage

```
BiplotCCA(
  x,
  ncol = 2,
  nrow = 1,
  jpeg = FALSE,
  select = FALSE,
  colour = "Spectral",
  xlim = NULL,
  ylim = NULL,
  max_cells_displayed = 30000,
  verbose = FALSE
)
```

Arguments

`x`	A TockyPrepData object produced by the function Tocky
`ncol`	The number of columns in plot
`nrow`	The number of rows in plot
`jpeg`	A logical argument. If FALSE, it will open a device window in which plots are generated.
`select`	Whether to interactively select markers to be processed. If FALSE, all the log-transformed markers apart from Timer fluorescence (stored in the TockyPrepData) will be used.
`colour`	Either 'Spectral' or 'BlueRed' for Angle colour key.
`xlim`	Optional x-axis limits.
`ylim`	Optional y-axis limits.
`max_cells_displayed`	The maximum number of cells to display in the plots
`verbose`	Logical indicating whether to print progress messages and outputs.

Value

Generates a CCA heatmap plot and optionally saves it as a jpeg file.

Examples

```
## Not run:
BiplotCCA(x)

## End(Not run)
```

`DijkstraTockyPath` *Dijkstra-Tocky Pathfinding*

Description

This function identifies paths within a Tocky Network using Dijkstra-Tocky Pathfinding

Usage

```
DijkstraTockyPath(x, origin_node, destination_node, verbose = TRUE)
```

Arguments

`x`	A TockyPrepData containing the network and relevant data.
`origin_node`	The starting node (cluster) for the path.
`destination_node`	The ending node (cluster) for the path.
`verbose`	Logical indicating whether to print progress messages and outputs.

Details

The function computes mean TimerAngles for each cluster, identifies all possible increasing paths from the origin to the destination, and then calculates the total weight for each path to find the closest one. Paths are determined by increasing TimerAngle values, ensuring that each step along a path moves to a node with a higher TimerAngle.

Value

The original TockyPrepData is returned with an additional list attached containing the ordered paths and the closest path.

Examples

```
## Not run:
x <- DijkstraTockyPath(x, origin_node = '3', destination_node = '21')

## End(Not run)
```

GetStatsClusterPercentage *Retrieve Cluster Percentage Data and Stats*

Description

This function retrieves the cluster percentage data from a TockyPrepData that has already been processed with the PlotClusterPercentage function. It can display the statistics in the Terminal or write them to CSV files.

Usage

```
GetStatsClusterPercentage(
  x,
  writeResults = FALSE,
  filename = "cluster_percentage"
)
```

Arguments

x	A TockyPrepData that has been processed with the PlotClusterPercentage function.
writeResults	A logical value. If TRUE, two files will be generated containing group statistics and p-values, respectively. If FALSE, these statistical results are displayed in the Terminal.
filename	(optional) Base name for the output files when writeResults is TRUE.

Value

The same TockyPrepData passed as input, for consistency in function design, though the function primarily focuses on data retrieval and display or file writing.

Examples

```
## Not run:
  GetStatsClusterPercentage(x)

## End(Not run)
```

PlotClusterPercentage *Plot Cluster Percentage by Bar plot*

Description

Plot Cluster Percentage by Bar plot

Usage

```
PlotClusterPercentage(
  x,
  p_adjust_method = "fdr",
  colours = NULL,
  verbose = TRUE
)
```

Arguments

x	A TockyPrepData after running the function ClusteringPCA.
p_adjust_method	A method for p-value adjustment in statistical tests.
colours	(optional) A vector specifying colours for different groups in the plot. For example, colours = c("purple", "black").
verbose	Logical indicating whether to print progress messages and outputs.

Value

The TockyPrepData with added statistics and plots for significant clusters.

Examples

```
## Not run:
x <- PlotClusterPercentage(x)

## End(Not run)
```

PlotDimRedLoading *Set a gate to define negative (and positive) for each marker expression*

Description

Set a gate to define negative (and positive) for each marker expression

Usage

```
PlotDimRedLoading(x, reduction = "PCA")
```

Arguments

x	A TockyPrepData.
reduction	Choose whether to use PCA or CCA as a reduction method.

Value

A TockyPrepData (unchanged) for safety.

Examples

```
## Not run:
PlotDimRedLoading(x)

## End(Not run)
```

`PlotPCAHeatmap` *Generate PCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)*

Description

Generate PCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)

Usage

```
PlotPCAHeatmap(
  x,
  ncol = 2,
  nrow = 1,
  jpeg = FALSE,
  select = FALSE,
  biplot_scaling = 3,
  colour = "Spectral",
  xlim = NULL,
  ylim = NULL
)
```

Arguments

`x`	A TockyPrepData object produced by the function Tocky
`ncol`	The number of columns in plot
`nrow`	The number of rows in plot
`jpeg`	A logical argument. If FALSE, it will open a device window in which plots are generated.
`select`	Whether to interactively select markers to be processed. If FALSE, all the log-transformed markers apart from Timer fluorescence (stored in the TockyPrepData) will be used.
`biplot_scaling`	A number for multiplying biplot values for visibility. Default is 3.
`colour`	Either 'Spectral' or 'BlueRed' for Angle colour key.
`xlim`	Optional x-axis limits.
`ylim`	Optional y-axis limits.

Value

Generates a CCA heatmap plot and optionally saves it as a jpeg file.

Examples

```
## Not run:
PlotPCAHeatmap(x)

## End(Not run)
```

`PlotTockyCCA` *Generate CCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)*

Description

Generate CCA heatmap plots for Tocky data (Timer-Blue vs Timer-Red 2d plots)

Usage

```
PlotTockyCCA(x, ncol = 4, nrow = 3, jpeg = FALSE, select = FALSE)
```

Arguments

`x`	A TockyPrepData object produced by the function Tocky
`ncol`	The number of columns in plot
`nrow`	The number of rows in plot
`jpeg`	A logical arguement. If FALSE, it will open a device window in which plots are generated.
`select`	Whether to interactively select markers to be processed. If FALSE, all the log-transformed markers apart from Timer fluorescence (stored in the TockyPrepData) will be used.

Examples

```
## Not run:
PlotTockyCCA(x)

## End(Not run)
```

`PlotTockyClustering` *Plot Tocky Clusters*

Description

Plot Tocky Clusters

Usage

```
PlotTockyClustering(
  x,
  jpeg = FALSE,
  max_cells_displayed = 30000,
  filename = NULL
)
```

Arguments

x	A TockyPrepData after running the function TockyClustering This function will generate PCA plots with cluster and Angle data
jpeg	Whether to out a jpeg file. The default is pdf = FALSE, by which a jpeg file is produced.
max_cells_displayed	The number of cells displayed in plots.
filename	A character string for file name

Value

The slot Reduction will contain the new slot Tocky_clusters, which includes kmeans clustering result

Examples

```
## Not run:
PlotTockyClustering(x)

## End(Not run)
```

PlotTockyNetwork *Plot A Tocky Network*

Description

Plot A Tocky Network

Usage

```
PlotTockyNetwork(
  x,
  reduction = "CCA",
  select_variable = FALSE,
  edge_scale = NULL,
  mds = FALSE,
  reflect = NULL,
  log2fc = FALSE,
  p_adjust = NULL
)
```

Arguments

x	A TockyPrepData after running the function TockyNetwork.
reduction	Choose whether to use PCA or CCA as a reduction method.
select_variable	Whether to choose a variable (marker expression). The default is FALSE and produces Tocky Angle and Intensity.
edge_scale	A scale factor for edge thickness.
mds	If Multidimensional Scaling is used for constructing network graph.

reflect	If Multidimensional Scaling is used, graph can be refelected by either 'horizontal' or 'vertical'.
log2fc	Logical. If TRUE, the colours of nodes are determined by log2 fold change between the two groups.
p_adjust	If p_adjust is "none", p-value adjustment is not used. Note that p-values from two-group comparisons are stored within the TockyPrepData object and have been calculated by 'PlotClusterPercentage'.

Value

Network plot is produced, showing clusters as vertices and distance between clusters as edges.

Examples

```
## Not run:
PlotTockyNetwork(x)

## End(Not run)
```

PlotTockyPCA *Produce PCA plot*

Description

Produce PCA plot

Usage

```
PlotTockyPCA(x, jpeg = FALSE, cluster = FALSE, filename = NULL)
```

Arguments

x	A TockyPrepData after running the function Tocky.
jpeg	A logical arguement. If FALSE, it will open a device window in which plots are generated.
cluster	A logical arguement. If TRUE, clusters are coloured in output plots.
filename	A character string for file name

Value

A TockyPrepData simply to avoid null return.

Examples

```
## Not run:
PlotTockyPCA(x)

## End(Not run)
```

plot_color_code *To produce colour key as a gradient rectangle*

Description

To produce colour key as a gradient rectangle

Usage

```
plot_color_code(
  expression = NULL,
  x,
  y,
  width,
  height,
  colour = "Spectral",
  method = "Expression"
)
```

Arguments

expression	A numeric vector defining the expression to be plotted.
x	A number. The x position of the lower left corner of the rectangle
y	A number. The y position of the lower left corner of the rectangle
width	A number. The width of the rectangle
height	A number. The height of the rectangle
colour	Either 'Spectral' or 'BlueRed' for Angle colour key.
method	Either 'Angle' or 'Spectral' for Angle and any expression, respectively.

Examples

```
## Not run:
plot(1:5)
plot_color_code(x = 3, y = 4.5, width = 1, height = 0.3, colour = 'Spectral', method = 'Angle')

## End(Not run)
```

showDijkstraTockyPath *Display Closest Tocky Angle Gradient Path*

Description

Retrieves and displays the closest Tocky Angle Gradient Path from a TockyPrepData. This function is intended for use after running DijkstraTockyPath to quickly access and review the computed closest path based on increasing TimerAngle values.

Usage

```
showDijkstraTockyPath(x, origin_node, destination_node)
```

Arguments

x	A TockyPrepData that has already been processed with DijkstraTockyPath to compute the Tocky Angle Gradient Path.
origin_node	The starting node (cluster) for the path.
destination_node	The ending node (cluster) for the path.

Value

Prints the closest path and also returns it for further analysis or usage.

Examples

```
## Not run:
showDijkstraTockyPath(x)

## End(Not run)
```

TockyCCA *Perform Dimensional Reduction using Canonical Correspondence Analysis (CCA) for TockyPrepData.*

Description

Perform Dimensional Reduction using Canonical Correspondence Analysis (CCA) for TockyPrepData.

Usage

```
TockyCCA(x, variables = NULL, marker_neg_gate = TRUE, select = FALSE)
```

Arguments

x	A TockyPrepData after running the function Tocky.
variables	Variables (markers) for CCA analysis. If NULL, variables are to be chosed interactively.
marker_neg_gate	Whether autofluorescence values are all considered to be zero or not. This approach is recommended. Perform DefineNegative for the same variables to be used by TockyCCA in advance.
select	Whether to interactively select markers to be processed. If FALSE, all the log-transformed markers apart from Timer fluorescence (stored in the TockyPrepData) will be used.

Value

The slot Reduction will contain CCA results.

Examples

```
## Not run:
x <- TockyCCA(x)

## End(Not run)
```

TockyClustering *Clustering Cells Using Dimensional Reduction by TockyPCA.*

Description

Clustering Cells Using Dimensional Reduction by TockyPCA.

Usage

```
TockyClustering(
  x,
  choose_dimension = FALSE,
  max_cells_displayed = 30000,
  num_centre = FALSE
)
```

Arguments

x	A TockyPrepData after running the function TockyPCA.
choose_dimension	Whether the number of dimension is specified by PCA plot. Default is FALSE.
max_cells_displayed	The number of cells displayed in plots for interatcive PCA plots.
num_centre	Whether the number of clusters is specified. Default is FALSE and the number of variables will be used.

Value

The slot Reduction will contain the new slot PCAclusters, which includes kmeans clustering result

Examples

```
## Not run:
x <- TockyClustering(x)

## End(Not run)
```

TockyNetwork *Produce A Network of Tocky Clusters*

Description

Produce A Network of Tocky Clusters

Usage

```
TockyNetwork(x, reduction = "CCA", cut_off = 0.2)
```

Arguments

x	A TockyPrepData after running the function TockyClustering.
reduction	Choose whether to use PCA or CCA as a reduction method. When using CCA, TockyCCA must have been applied to the TockyPrepData.
cut_off	Threshold value as a quantile percentage for edge connection. For example, the default 0.2 will set the threshold at 20% quantile of distance between clusters, connecting neighbor clusters.

Value

A revised TockyPrepData containing an igraph network object. Network is constructed based on the neighbor proximity.

Examples

```
## Not run:
x <- TockyNetwork(x)

## End(Not run)
```

TockyPCA *Perform Principal Component Analysis for TockyPrepData.*

Description

Perform Principal Component Analysis for TockyPrepData.

Usage

```
TockyPCA(
  x,
  variables = NULL,
  marker_neg_gate = TRUE,
  cleaning = FALSE,
  select = TRUE,
  Timer = FALSE
)
```

Arguments

x	A TockyPrepData after running the function Tocky.
variables	Variables (markers) for PCA analysis. If NULL, variables are to be chosed interactively.
marker_neg_gate	Whether autofluorescence values are all considered to be zero. This approach is recommended and requires DefineNegative.
cleaning	Whether data cleaning is performed to remove cells with a zero value for a marker. This is not recommended unless you have a reason that you cannot use DefineNegative to collapse negative data.
select	Whether to interactively select markers to be processed. If FALSE, all the log-transformed markers apart from Timer fluorescence (stored in the TockyPrepData) will be used.
Timer	Whether Timer data (normalised or Angle/Intensity) are used.

Examples

```
## Not run:
x <- TockyPCA(x, variables = NULL, cleaning = FALSE, marker_neg_gate = TRUE)

## End(Not run)
```

Index